Università degli Studi di Salerno
Corso di Ingegneria del Software

W3Play
ODD
Versione 1.1

Data:

Partecipanti:

Nome	Matricola
Luca Anzalone	0512103488
Augusto D'Alessio	0512103752
Alfonso Di Pace	0512103608
Antonio Vivone	0512103650

Revision History

Data	Versione	Descrizione	Autore
14/12/2017	0.1	Prima stesura	Tutti
10/01/2018	1.0		Tutti
02/02/2018	1.1	Aggiunti diagrammi packages	Luca Anzalone

Sommario

1. Introduzione

1. Object Design Trade-offs

Dopo la realizzazione dei documenti RAD e SDD abbiamo descritto in linea di massima quello che sarà il nostro sistema e quindi i nostri obiettivi, tralasciando gli aspetti implementativi. Il seguente documento ha lo scopo di produrre un modello capace di integrare in modo coerente e preciso tutte le funzionalità individuate nelle fasi precedenti. In particolare, definisce le interfacce delle classi, le operazioni, i tipi, gli argomenti e le signature dei sottosistemi definiti nel System Design. Inoltre, sono specificati i trade-off e le linee guida.

Comprensibilità vs Tempo:
Il codice deve essere quanto più comprensibile possibile per facilitare la fase di testing ed eventuali future modifiche. Il codice sarà quindi accompagnato da commenti che ne semplifichino la comprensione. Ovviamente questa caratteristica aggiungerà un incremento di tempo allo sviluppo del nostro progetto.

Interfaccia vs Usabilità:
L'interfaccia grafica è stata realizzata in modo da essere molto semplice, chiara e concisa, fa uso di form e pulsanti disposti in maniera da rendere semplice l'utilizzo del sistema a quanti più utenti possibili.

Sicurezza vs Efficienza:
La sicurezza, come descritto nei requisiti non funzionali del RAD, rappresenta uno degli aspetti importanti del sistema. Tuttavia, dati i tempi di sviluppo molto limitati, ci limiteremo ad implementare sistemi di sicurezza basati su username e password degli utenti.

Response Time vs Hardware:
Il sistema garantisce una certa reattività alle richieste, e quindi essere in grado di poter comunque offrire una contemporaneità di servizi agli utenti. Ovviamente questa caratteristica sarà limitata dall'hardware del sistema.

Prestazioni vs Costi
Il sistema prevede l'utilizzo di template open source esterni per mantenere prestazioni elevate, essendo il progetto sprovvisto di budget.

2. Linee Guida per la Documentazione delle Interfacce

Gli sviluppatori dovranno seguire le seguenti convenzioni per la scrittura del codice:

Naming Convention

- È buona norma utilizzare nomi:
  - Descrittivi
  - Pronunciabili
  - Di uso comune
  - Lunghezza medio-corta
  - Non abbreviati
  - Evitando la notazione ungara
  - Utilizzando solo caratteri consentiti (a-z, A-Z, 0-9)

Variabili

- I nomi delle variabili devono cominciare con una lettera minuscola (es: prodotto). Se il nome della variabile è costituito da più parole, solo l'iniziale delle altre parole sarà maiuscola (es: codiceFiscale), inoltre è possibile abbreviare il nome della variabile solo se non peggiora la leggibilità e comprensibilità (es: numArticoli invece di numeroArticoli).
- Le costanti dovranno essere scritte interamente in maiuscolo e se il nome è costituito da più parole vengono separate con l'underscore (es: PI_GRECO).

Metodi

- I nomi dei metodi devono cominciare con una lettera minuscola, e le parole seguenti con la lettera maiuscola. Il nome del metodo tipicamente consiste di un verbo che identifica una azione, seguito dal nome di un oggetto.
- I nomi dei metodi per l'accesso e la modifica delle variabili dovranno essere del tipo getNomeVariabile() e setNomeVariabile().
- I commenti dei metodi devono essere raggruppati in base alla loro funzionalità, la descrizione dei metodi deve apparire prima di ogni dichiarazione di metodo, e deve descriverne lo scopo. Deve includere anche informazioni sugli argomenti, sul valore di ritorno, e se applicabile, sulle eccezioni.

Classi e pagine

- I nomi delle classi e delle pagine devono iniziare con una lettera maiuscola, le parole contenute al suo interno devono cominciare con lettera maiuscola.
  Il nome deve fornire informazioni utili relative al loro scopo.

Ogni file sorgente contiene una singola classe e deve essere strutturato in un determinato modo:

- L' istruzione package che permette di inserire la classe in un determinato package
- L' istruzione import che importa le librerie necessarie alla class
- Una piccola descrizione della classe

L’ introduzione mostra l’autore e una descrizione.
ESEMPIO:

```
/**
 * descrizione della classe
 * @author nome dell’utente
 */
```

- La dichiarazione di classe caratterizzata da:
    1. Dichiarazione della classe pubblica
    2. Costruttore
    3. Metodi
    4. Dichiarazioni di costanti, variabili di classe e di istanza

ESEMPIO:

```
public class Ordine{

/**
 * @param idOrdine Id dell’ ordine
 * @param costo Costo dell’ ordine
 * @param dataAcquisto Data di acquisto dell’ ordine
 * @param carta Carta utilizzata nell’ ordine
 * @param utente Utente che ha fatto l’ ordine
 */
public Ordine(int idOrdine, float costo, LocalDate dataAcquisto,String
carta, String utente)
{
      this.idOrdine = idOrdine;
      this.costo = costo;
      this.dataAcquisto = dataAcquisto;
      this.carta = carta;
      this.utente = utente;
}

/**
 * @return int L’ Id dell’ ordine
 */
public int getIdOrdine() {
      return idOrdine;
}

/**
 * Setta l’ ID dell’ ordine
 * @param idOrdine
 */
public void setIdOrdine(int idOrdine) {
      this.idOrdine = idOrdine;
}

private  int idOrdine;
private float costo;
private  String carta, utente;
private  LocalDate dataAcquisto;
private ArrayList<OggettoOrdine> articoli;

}
```

1.3 Definizioni, acronimi e abbreviazioni

Acronimi

- RAD: Requirements Analysis Document
- SDD: System Design Document
- ODD: Object Design Document
- DBMS: Database Management System
- API: Application Programming Interface
- JDBC: Java Database Connectivity

2. Packages

Il sistema è diviso in 3 livelli:

- Interface Layer
- Application Logic Layer
- Storage Layer

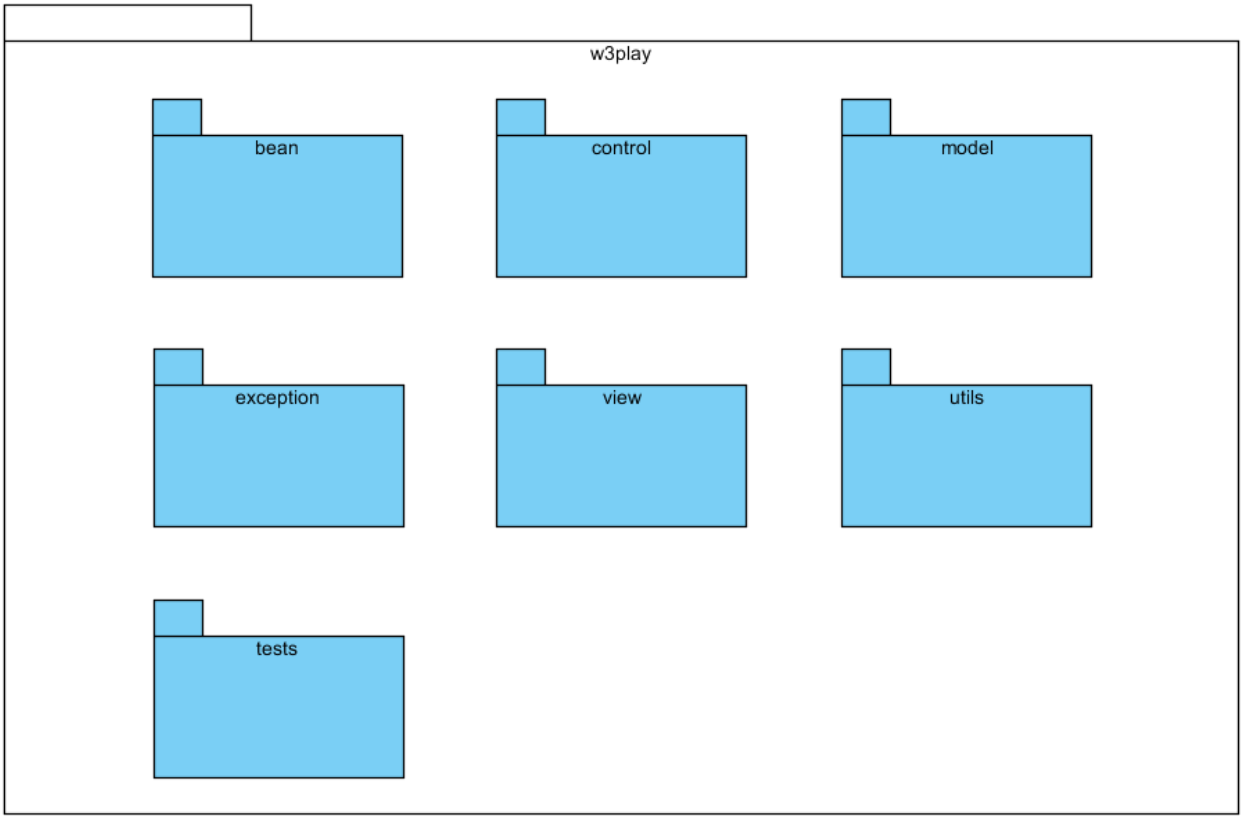
I sottopackage del sistema contengono classi atte allo svolgimento delle richieste pervenute dall'utente, sia per gestirne la logica sia per offrire risposte sotto forma di interfaccia grafica.

Interface Layer	Rappresenta l'interfaccia del sistema, ed offre la possibilità all'utente di interagire con quest'ultimo, offrendo sia la possibilità di inviare, in input, che di visualizzare, in output, dati.

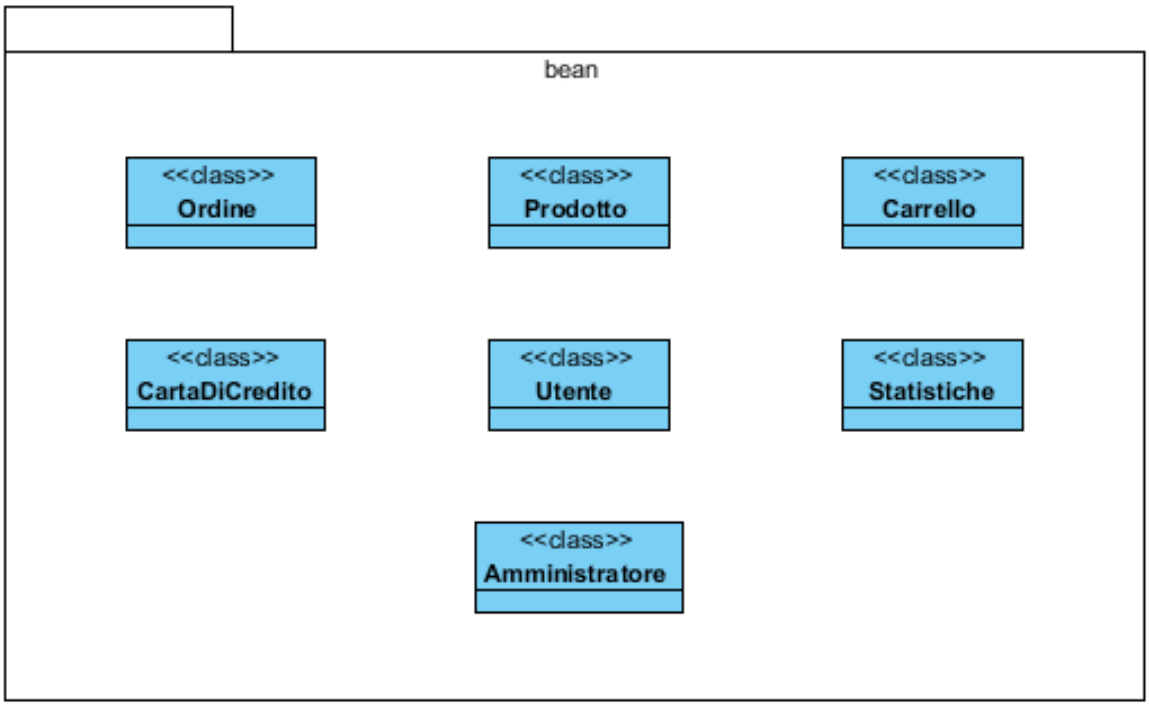
Application Logic Layer	Si occupa dell'elaborazione dei dati da inviare al client. Si occupa di varie gestioni quali: 1. Gestione Utente 2. Gestione Amministratore 3. Gestione Carrello 4. Gestione Ordine 5. Gestione Prodotto

Storage Layer	Ha il compito di memorizzare i dati sensibili del sistema, utilizzando un DBMS che gestisce i dati. Inoltre, lo Storage Layer riceve le varie richieste dall' Application Logic Layer inoltrandole al DBMS e restituendo i dati richiesti. La comunicazione verso il DBMS è realizzata tramite le API JDBC.

2.1.Package w3play

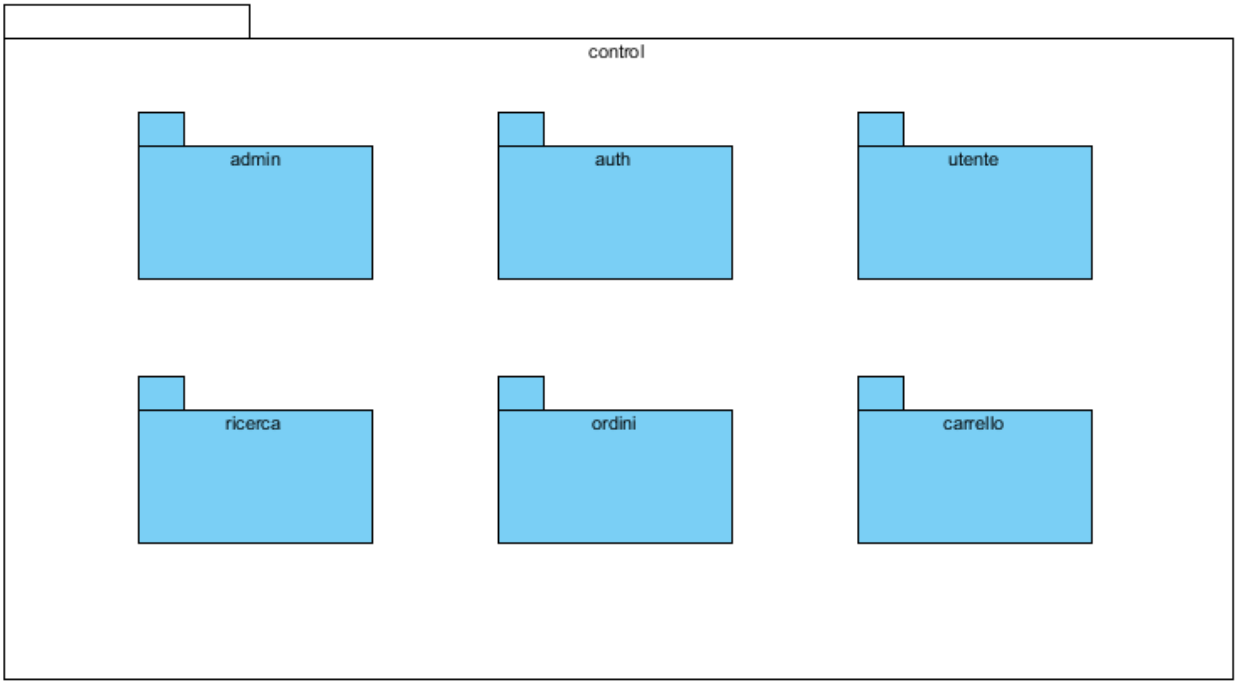


2.2. Package Bean

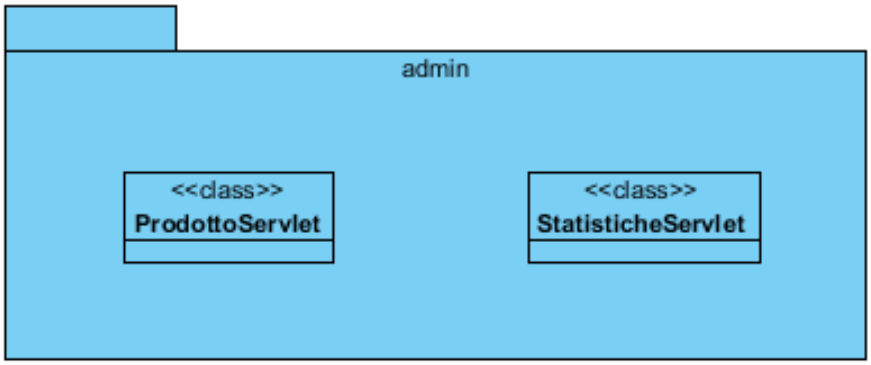


Class	Descrizione
Ordine.java	Descrive un ordine effettuato dai clienti.
Prodotto.java	Descrive un prodotto presente nel catalogo
Carrello.java	Descrive il carrello posseduto dall'utente
CartaDiCredito.java	Descrive la carta di credito inserita dall'utente come metodo di pagamento
Utente.java	Descrive l'utente
Statistiche.java	Descrive le statistiche inerenti al negozio online
Amministratore.java	Descrive l'amministratore del sistema

2.3.Package Control

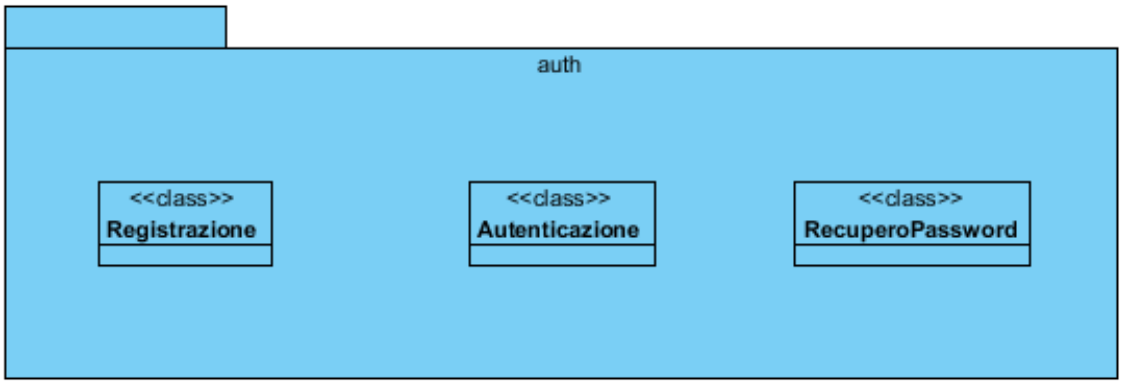


2.3.1. Package Admin



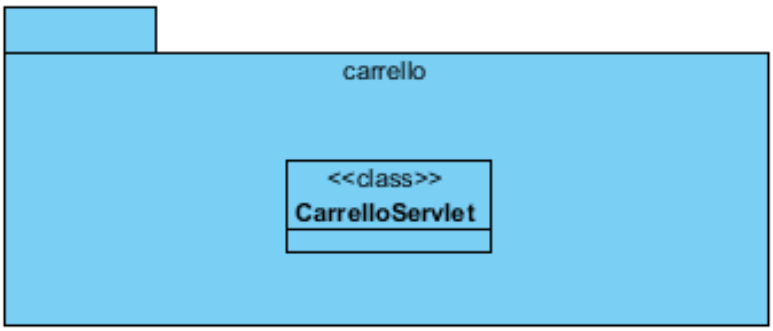
Classe	Descrizione
ProdottoServlet.java	Servlet per le operazioni dell'admin
StatisticheServlet.java	Servlet per il calcolo delle statistiche

2.3.2. Package Auth



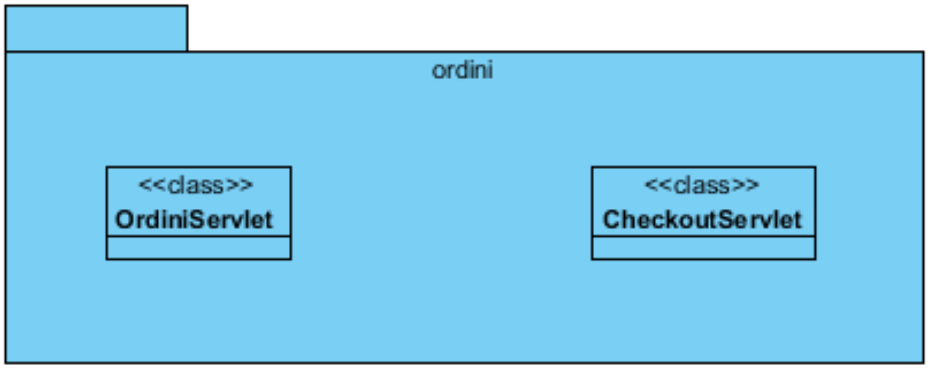
Classe	Descrizione
Autenticazione.java	Servlet che gestisce l'autenticazione
Registrazione.java	Servlet che gestisce la registrazione
RecuperoPassword.java	Servlet che gestisce il recupero password

2.3.3. Package Carrello



Classe	Descrizione
CarrelloServlet.java	Servlet che gestisce il carrello

2.3.4. Package Ordini

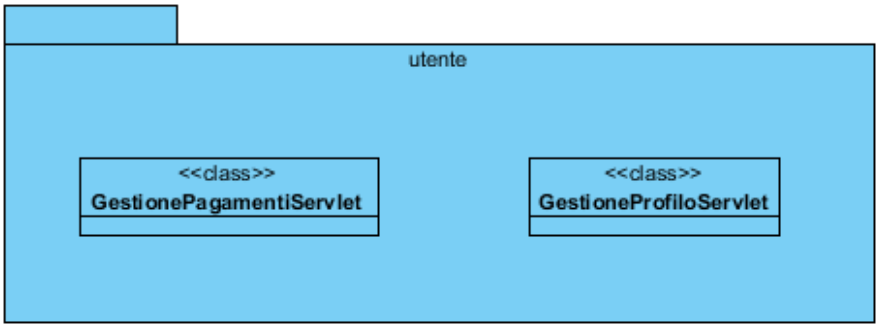


Classe	Descrizione
OrdiniServlet.java	Servlet che gestisce ordini già effettuati
CheckoutServlet.java	Servlet che crea ordini

2.3.5. Package Ricerca

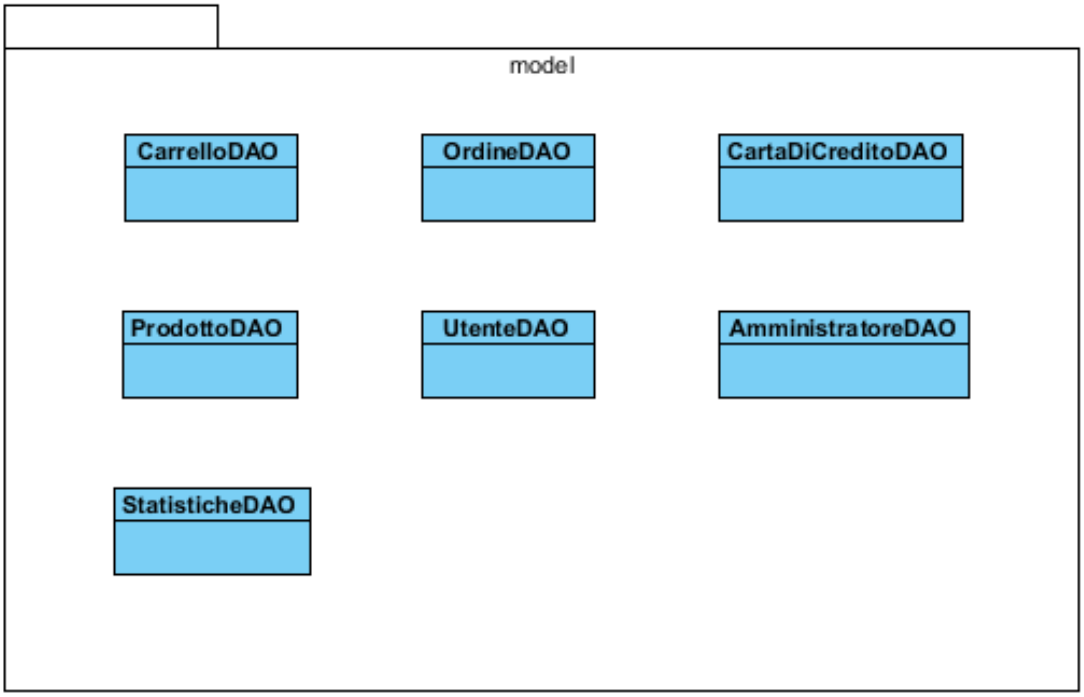
Classe	Descrizione
RicercaServlet.java	Servlet che gestisce la ricerca

2.3.6. Package Utente



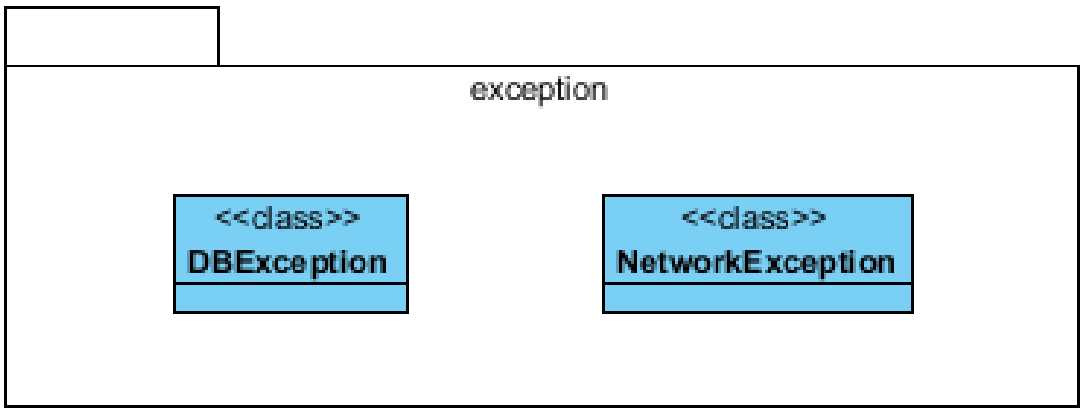
Classe	Descrizione
GestioneProfiloServlet.java	Servlet per le modifiche del profilo
GestionePagamentiServlet.java	Servlet per la gestione pagamenti

2.4.Package Model



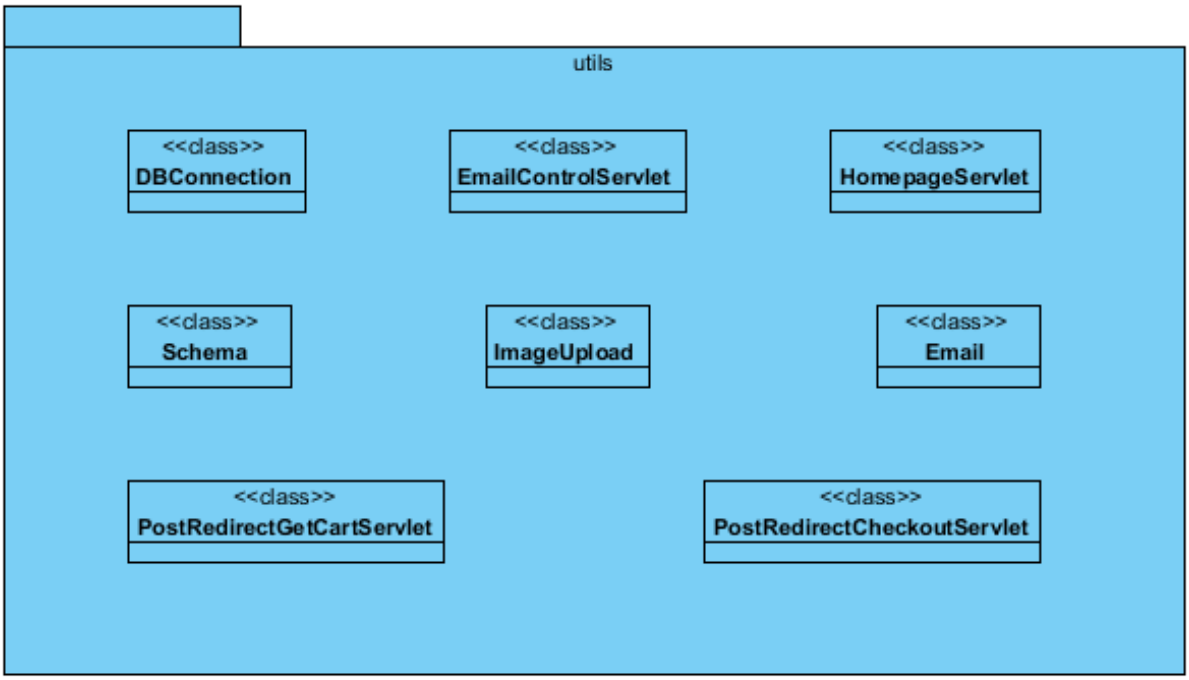
Classe	Descrizione
CarrelloDAO.java	Descrive l'accesso al database relativo all'oggetto carrello.
OrdineDAO.java	Descrive l'accesso al database relativo all'oggetto ordine.
CartaDiCreditoDAO.java	Descrive l'accesso al database relativo all'oggetto carta di credito.
ProdottoDAO.java	Descrive l'accesso al database relativo all'oggetto prodotto.
UtenteDAO.java	Descrive l'accesso al database relativo all'oggetto utente.
AmministratoreDAO.java	Descrive l'accesso al database relativo all'oggetto amministratore.
StatisticheDAO.java	Descrive l'accesso al database relativo all'oggetto statistiche.

2.5. Package Exception



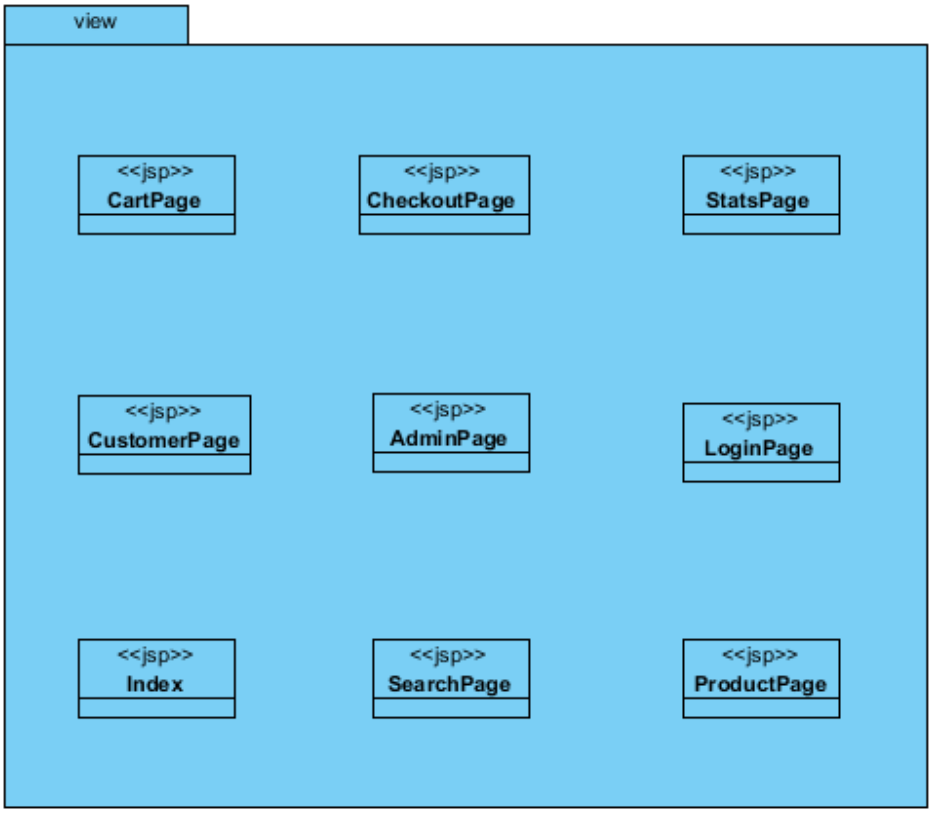
Class	Descrizione
DBException.java	Indica errori inerenti al database con relativi messaggi
NetworkException.java	Indica errori con la connessione

2.6. Package Utils



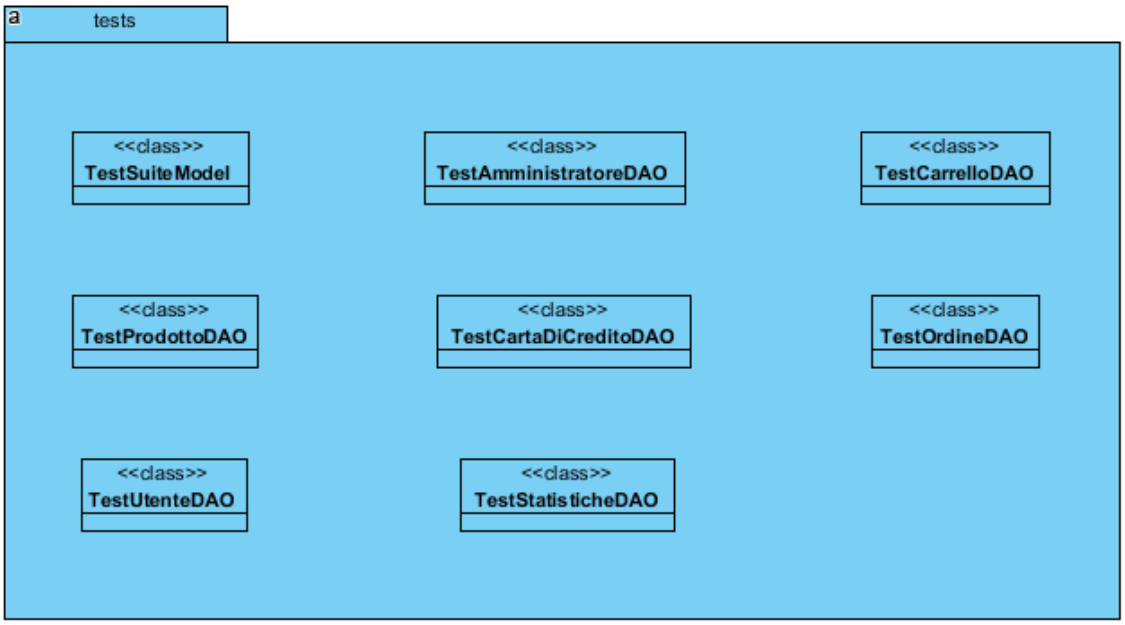
Classe	Descrizione
DBConnection.java	Servlet per la connessione al database
Email.java	Classe per l'invio della e-mail di recupero
EmailControlServlet.java	Servlet per il controllo della e-mail
HomepageServlet.java	Servlet per la gestione della homepage
ImageUpload.java	Servlet per il caricamento di immagini
PostRedirectGetCartServlet.java	Servlet per realizzare pattern POST/REDIRECT/GET relativo al Carrello
PostRedirectGetCheckoutServlet.java	Servlet per realizzare pattern POST/REDIRECT/GET relativo al Checkout
Schema.java	Contiene stringhe che rappresentano le colonne delle tabelle del database

2.7.Package View



Classe	Descrizione
CartPage.jsp	Pagina per permettere all'utente di visualizzare e gestire il carrello
CheckoutPage.jsp	Pagina per l'acquisto dei prodotti presenti nel carrello
CustomerPage.jsp	Pagina per la gestione del profilo utente
AdminPage.jsp	Pagina per la gestione del profilo dell'utente amministratore
LoginPage.jsp	Pagina per la registrazione, login e recupero password
Index.jsp	Homepage del Sistema
StatsPage.jsp	Pagina per la visualizzazione delle statistiche di sistema
SearchPage.jsp	Pagina che contiene i prodotti trovati in base alla query di ricerca immessa dall'utente
ProductPage.jsp	Pagina dettaglio prodotto

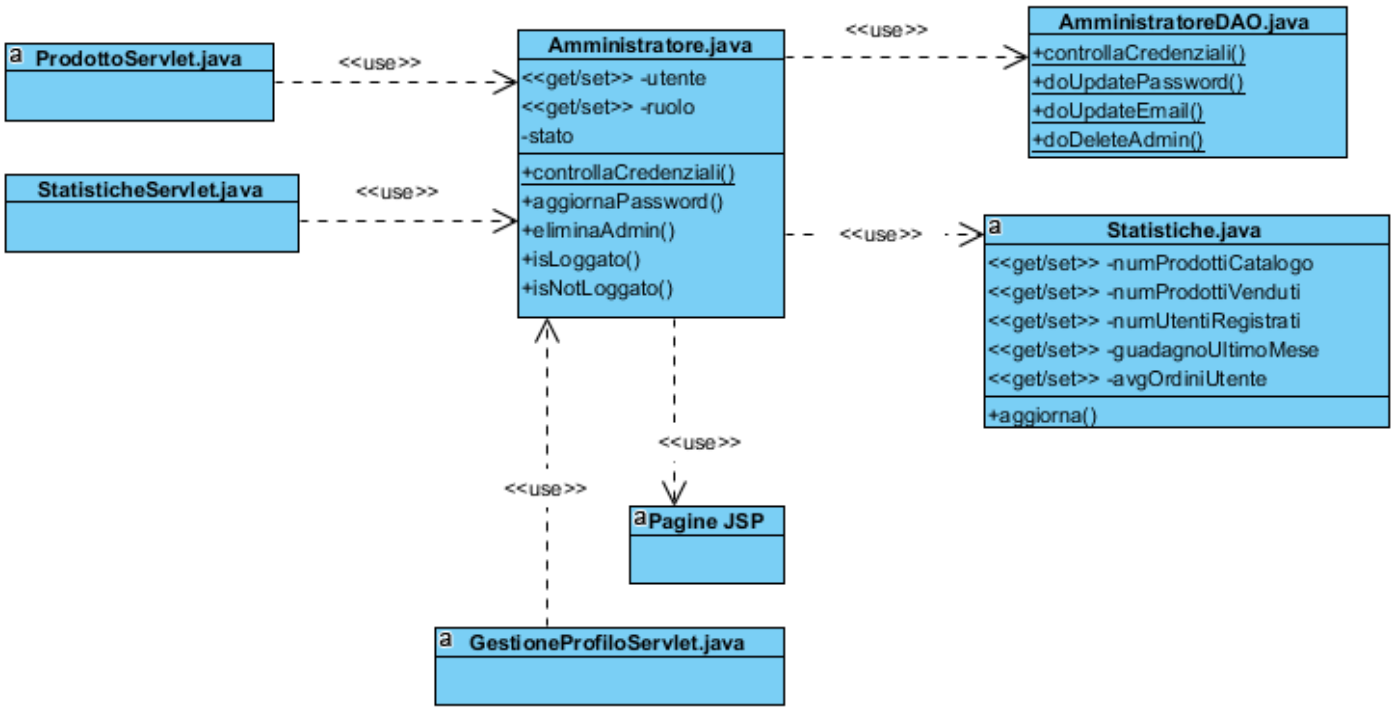
2.8. Package Tests



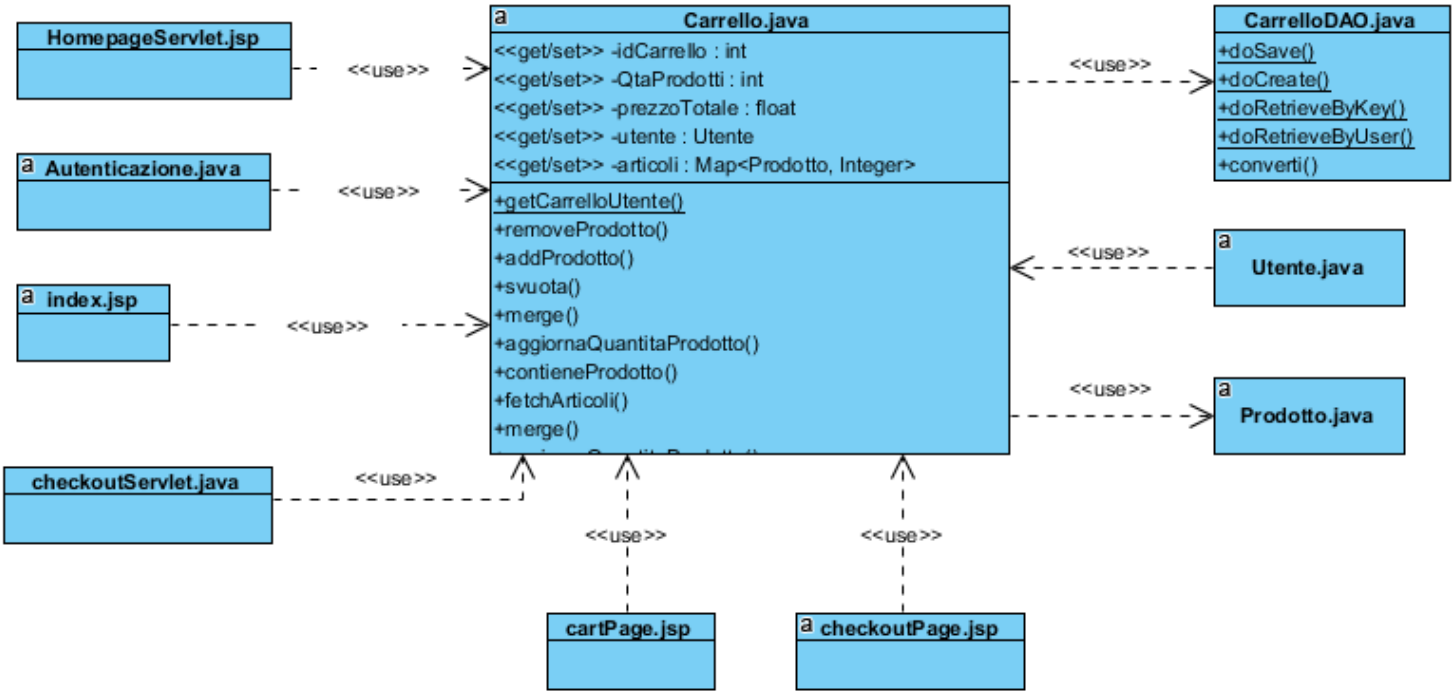
Classe	Descrizione
TestSuiteModel.java	Test Suite che include tutte le classi di test per il model
TestAmministratoreDAO.java	Classe di test per AmministratoreDAO
TestCarrelloDAO.java	Classe di test per CarrelloDAO
TestProdottoDAO.java	Classe di test per ProdottoDAO
TestCartaDiCreditoDAO.java	Classe di test per CartaDiCreditoDAO
TestOrdineDAO.java	Classe di test per OrdineDAO
TestUtenteDAO.java	Classe di test per UtenteDAO
TestStatisticheDAO.java	Classe di test per StatisticheDAO

3. Class Interface

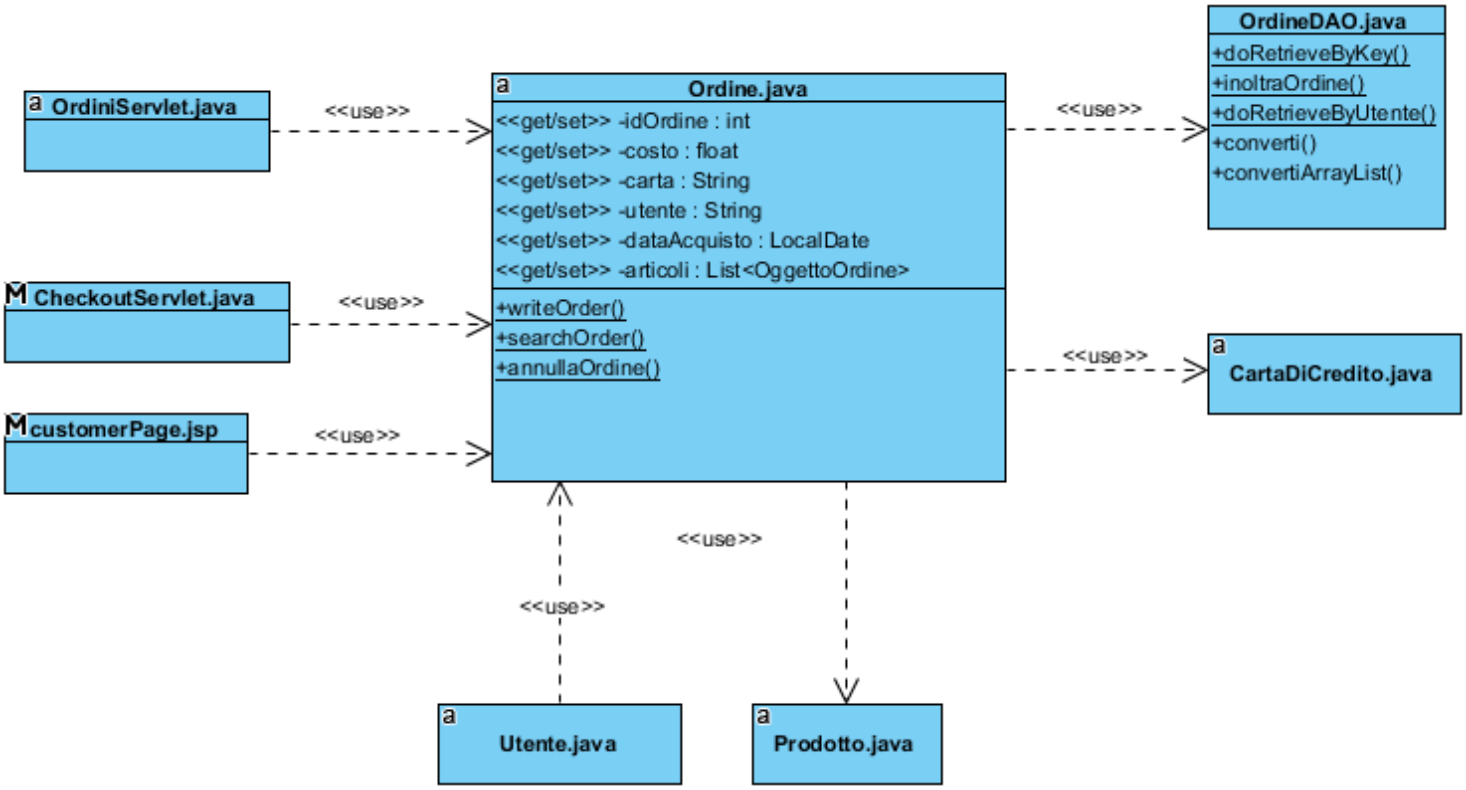
3.1 Gestione Amministratore



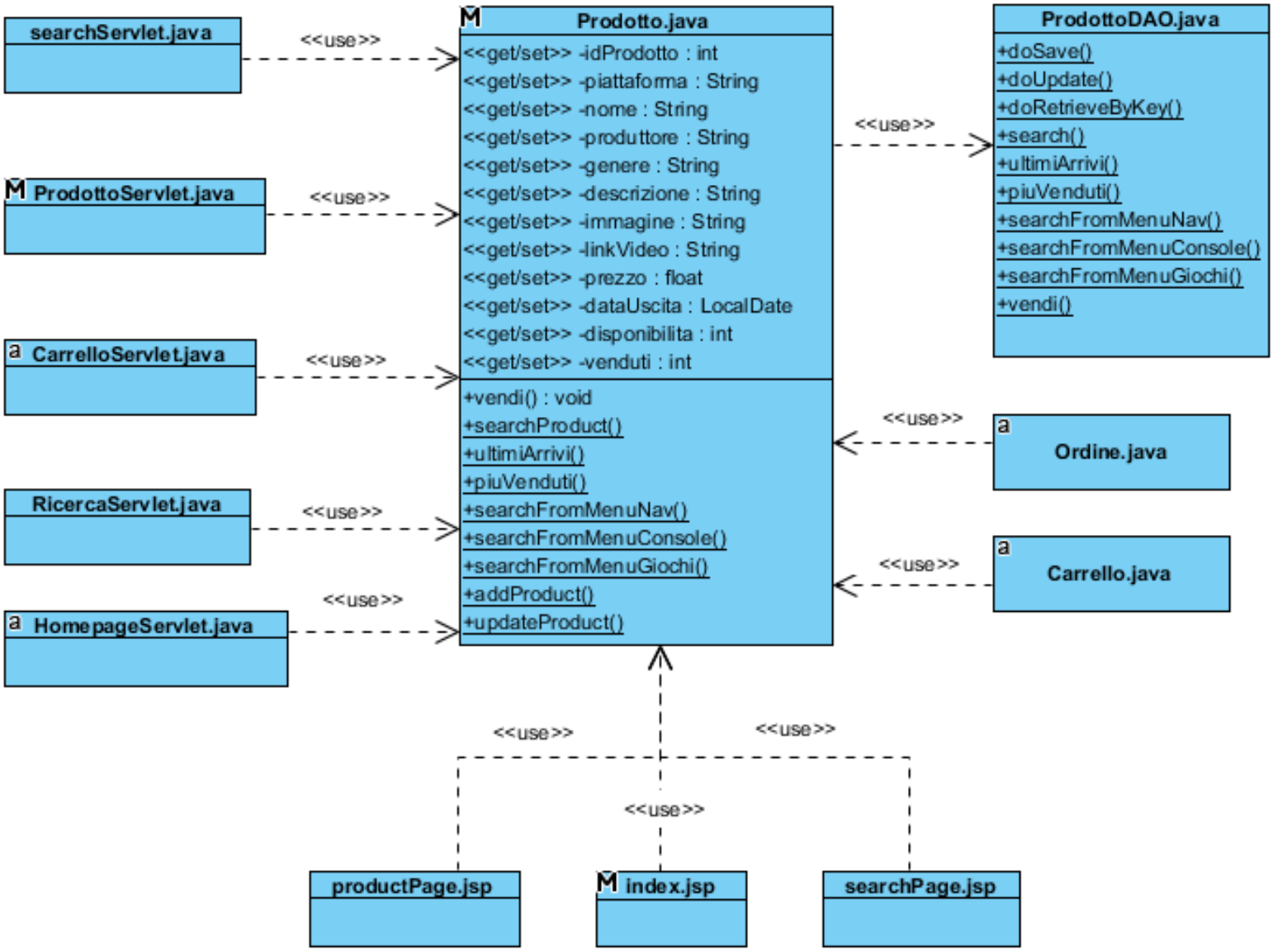
2. Gestione carrello



3. Gestione ordine



4. Gestione prodotto



5. Gestione utente